Layered Security - Gradient Boosting meets Naive Bayes for Intrusion Detection

Srinivasa Chakravarthy Kamireddypalle [1][0009-0007-1777-1900], Dr. G. Shyama Chandra Prasad [2][0000-0001-9223-3060], Krishna Sai Srinivas Vootla [3][0009-0004-4053-4848] and Piyush Bhuyan [4][0009-0005-5389-6044]

[1,3,4] Department of Information Technology, Vasavi College of Engineering, Ibrahimbagh, Hyderabad, Telangana, India
[2] Department of Computer Science and Engineering, Bhoj Reddy Engineering College for Women, Hyderabad, Vinay Nagar, Saidabad, Hyderabad , Telangana, India

1 ks.chakravarthy@staff.vce.ac.in
2 gscprasad@gmail.com
3 vkssrinivas03@gmail.com
4 mrpiyush0007@gmail.com

Abstract. While cyberattacks get increasingly more sophisticated and common, traditional Intrusion Detection Systems (IDS) usually lack the capacity to catch new kinds of attacks, especially infrequent ones like User to Root (U2R) and Remote to Local (R2L) attacks. This paper suggests an upgraded Double-Layered Hybrid Approach (DLHA) with the addition of Gradient Boosting (or, specifically, eXtreme Gradient Boosting or XGBoost) alongside Naive Bayes classifiers. This strategy substitutes Support Vector Machine (SVM) in the second layer with XGBoost and adds synthetic augmentation with Generative Adversarial Networks (GANs) to counter class imbalance. Experimental comparisons on the NSL-KDD dataset show better detection rates, particularly for sparse classes, as well as lower false alarm ratios and computational complexity. This architecture offers a scalable, precise, and effective solution to contemporary Intrusion Detection System (IDS) challenges.

Keywords: Double Layered Hybrid Approach, Support Vector Machine, Naive Bayes, eXtreme Gradient Boosting, Intrusion Detection System, Generative Adversarial Network.

1 Introduction

1.1 Background and Motivation

The growing reliance on interconnected systems in sectors like healthcare, finance, and defense has positioned cybersecurity as a critical issue. Intrusion Detection Systems (IDS) play an essential role in detecting unauthorized access or malicious activity on

networks. Conventional IDS are either Signature-Based Intrusion Detection Systems (SIDS) or Anomaly-Based Intrusion Detection Systems (AIDS).

SIDS work by matching network traffic against known patterns of malicious activity. They are quick and accurate for known threats but cannot detect new or changing attacks.

AIDS, however, detect deviations from learned normal behavior and are therefore able to detect zero-day attacks, but tend to have high false positive rates.

In both models, Machine Learning (ML)-based methods have proved to be viable alternatives that offer adaptive learning features and flexibility. Yet, ML models remain to tackle some fundamental challenges like:

- Imbalanced datasets (particularly for infrequent attacks such as R2L and U2R)
- High computational expense (e.g., in Support Vector Machine - SVM)
- Weak generalization over all attack modes

1.2 Background and Motivation

Infrequent attacks such as R2L and U2R tend to be classified as legitimate traffic because of their rarity in training data and behaviorally similar nature to normal activity. Support Vector Machine (SVM), though widely used in IDS, is computationally intensive and hyperparameter sensitive. Real-time and scalable deployment thus becomes difficult.

The objective of this paper is to solve these problems by suggesting a hybrid model with a layered architecture that:

- Utilizes Naive Bayes (NB) for identifying frequent attack types (Denial of Service - DoS, and Probe).
- Utilizes XGBoost for recognizing rare attack types (R2L and U2R).
- Utilizes GANs to create synthetic data for underrepresented classes.

2 Literature Review

Network intrusion detection systems (NIDS) have come a long way with the incorporation of sophisticated machine learning approaches. The following literature survey discusses existing work with specific emphasis on hybrid models, dimension reduction methods, and approaches solving class imbalance problems.

2.1 Hybrid and Ensemble Approaches for Intrusion Detection

Mora-Gimeno et al. [1] proposed a two-stage modular intrusion detection system design that enhances detection performance significantly using varied concurrent detection methods. In the first stage, various intrusion detection mechanisms are used to capture the overall application behavior, which significantly increases detection probability, while the modular design provides flexibility and responsiveness. This multi-

detector mechanism introduces computational costs, degrading efficiency and scalability for practical implementations.

Wisanwanichthan et al. [12] introduced a double-layered hybrid method that integrates Naive Bayes and Support Vector Machine (SVM) [13] for network intrusion detection. Their system attained 88.97% accuracy, 90.57% F1 score, 88.17% precision, and 93.11% detection rate with an 11.82% false alarm ratio against the KDDTrain+ dataset. The method proved to work well even on a smaller dataset (KDDTrain+_20Percent) with relatively similar performance metrics. Their usage of Intersectional Correlated Feature Selection (ICFS) diminishes dimension and boosts efficiency but is still subject to the quality of training data, which can hamper adaptability to changing threats.

Single-model strategies have been shown in recent studies to be limited. Decision forests are the only intrusion detection systems that most AI-based intrusion detection systems, except for [15, 16], use and do not combine decisions for optimal IDS management. These models tend to have high false positive rates, leading to thousands of security alerts per day in big organizations [17], and also have a high risk of false negative rates [18]. This shortage indicates a requirement for ensemble learning techniques to enhance IDS performance [19, 20].

2.2 Optimization Algorithms and Feature Selection

Li et al. [2] proposed the Gravitational Search Algorithm (GSA), in which particles (mass objects) are drawn towards fitter ones with an attraction to move toward optimal solutions based on gravitational interaction without direction from the environment. GSA has been found efficient for feature selection by selecting relevant features based on its mass-based attraction operation. GSA has a drawback of slow convergence based on diminishing gravitational force in the course of subsequent iterations, especially for high-dimensional problem spaces. Also, overexploitation can lead to particles rushing prematurely towards local optima, preventing further searching towards global optima.

Siddiqi et al. [7] discussed how high-dimensional data harms anomaly detection rates in NIDS based on machine learning. Their work identified the inefficiency of transforming non-image network traffic data to image forms for anomaly detection. In addition, they showed that excessive dependence on large sets of features undermines real-time detection, an essential criterion for successful deployment of NIDS.

Pajouh et al. [8] proposed a two-layer dimension reduction and two-tier classification (TDTC) framework for intrusion detection. The method uses a dimension reduction module to deal with high-dimensional data, together with a two-tier classification module that utilizes Naive Bayes and a certainty factor variant of k-NN to identify abnormal activity. Although novel, the validation of the model was confined to the NSL-KDD dataset, which can be of concern regarding applicability to other datasets and real-world network settings.

2.3 Deep Learning Approaches

Deep learning techniques have proven to be very effective in intrusion detection. A deep learning-based IDS for detecting cyber-attacks on Controller Area Networks (CANs) with 99% accuracy using an LSTM autoencoder has been suggested in earlier work [3]. Nevertheless, one of the main shortcomings of LSTM-based methods is that they are sequential in nature, which disallows parallel input processing and results in longer detection times—a vital shortcoming for real-time threat detection.

Other methods have investigated sparse autoencoder-based NIDS [4, 5], with reported accuracy of 79.1% for multiclass classification using the NSL-KDD dataset. Nevertheless, the NSL-KDD dataset has been largely deemed outdated and plagued by class imbalance, restraining model generalizability and applicability in real-world scenarios. Certain methods try to use network data in a direct-to-feature manner by using one-hot encoding on every byte value in packet data [6], which gives rise to feature spaces larger than the original packet size, leading to computational inefficiencies.

2.4 Addressing Class Imbalance and Data Quality

Class imbalance remains a significant obstacle to intrusion detection. Akbani et al. [10] posit that Occam's razor dictates that the classifiers favor the majority class when the training data were skewed. This bias and its intent to develop classifiers degrade the performance of classifiers for the minority categories-a serious issue for security settings because rare attack varieties can be the most damaging.

Idhammad et al. [14] considered semi-supervised methods of DDoS detection where SMOTE is used to address imbalance by oversampling minority instances; their implementation replicated training set minority examples about 50–110 times. While excessive oversampling can reduce false alarm rates, aggravation by oversampling can also adversely affect overall detection performance and thus should be controlled.

Ahmed and Mahmood [11] proposed a hybrid anomaly detection framework for NIDS that combines both supervised and unsupervised learning. They note that purely unsupervised approaches often achieve low precision and high false positives in domains where labeled data are scarce. Their method decreases reliance on labels while exploiting unlabeled data to increase detection precision.

2.5 Dataset Challenges and Evaluation Methods

Dataset quality is a major factor governing NIDS effectiveness. Leevey and Khoshgoftaar [9] reviewed intrusion detection models based on CICIDS2018 and identified several problems: models tend to overfit when tested on the same dataset; class imbalance is usually underreported; and improper data-cleaning routines can affect, and thus degrade, IDS performance.

In light of advances in generative modeling, and especially GANs, the practical approach is to mitigate the dataset pitfalls by providing synthetic minority-class examples. This should rebalance class distributions while maintaining the characteristics of rare

attack types, thereby hopefully increasing detection rates for uncommon yet critical attacks [21, 22].

Literature review brings forth clear gaps: single-method solutions seem promising; however, they usually lack a wide robustness range, and this motivates the pursuit of hybrid approaches with complementary algorithms that will work through each other's shortcomings. Handling class imbalance better via sampling or advanced generative eterings remains key to detecting robust-rare attacks. And finally, proposals must be cross-validated on multiple recent datasets in order to prove generizability and operational potential.

3 Methodology

Our solution extends the Double-Layered Hybrid Architecture (DLHA) framework and proposes two innovations:

- Substitution of the second-layer SVM with XGBoost for enhanced detection of rare classes.
- Employment of Generative Adversarial Networks (GANs) to artificially create samples of rare attacks (R2L and U2R).

3.1 GAN Architecture

The vanilla GAN is the primal adversarial scheme with a generator transforming noise vectors into synthetic records with the discriminator learning to discern real from fake samples- they train opposing each other so that the generator slowly improves in its degree of realism. This model was chosen as the starting point since it is the most natural baseline for learning an entire joint distribution of tabular features without imposing manual assumptions, and it is straightforward to prototype on network intrusion logs. However, it brought to light on KDD critical flaws: severe class imbalance made the generator collapse onto majority modes and produce mostly normal and common attack types, while minority attacks were neglected; the discriminator would eventually exploit the low cardinality, high information categorical features to generate either vanishing or misleading gradients toward the rare regions of the space, so training would oscillate or converge to trivial output. Because the vanilla GAN offers no mechanism on request to get samples of a particular class and its optimization is fragile on skewed tabular data, it could not fulfill the objective of balanced synthetic attack generation.

The conditional GAN enhances the baseline setup by feeding class labels into both the generator and the discriminator so that the generator learns to produce samples conditioned on a requested label, while the discriminator judges samples depending on that label. Conditioning was deployed to directly tackle class imbalance: rather than hoping minority modes might be discovered on their own, the training process could repeatedly request and oversample particular rare attack classes, thus explicitly providing the model with supervision for those classes. Conditioning was successful in reinstating

sampling control and explicitly presented the generator with separate conditional mappings to cover generation for the targeted classes. However, cGANs still suffered from the underlying adversarial instability: mode collapse and weak convergence sometimes occurred even within individual conditioned classes because adversarial loss continued to be sensitive and very limited class-support made it difficult for the generator to learn strong intra-class distributions. Hence, cGAN really solved the sampling control problem but did not act as a reliable means to stabilize optimization when training examples per label were very scarce.

To stabilize training, the Wasserstein GAN was employed by replacing the original discriminator objective, whose critic, assigning a continuous score intended to represent distributional distance, ensures smooth gradient flow and more meaningful guidance to the generator. WGAN was chosen because its signal for learning remains meaningful even when the real and generated distributions are mutually exclusive-a situation that commonly arises with rarely occurring attacks-and because it tends to prevent collapse and makes hyperparameter sensitivity less of an issue. On KDD, WGAN was also found to produce more diverse samples and to help avoid catastrophic collapse, so that the generator could keep on making improvements even when the critic became quite strong. However, the practical approach taken to enforce the theoretical constraints necessitates the use of weight clipping-again, introducing a kind of fragility: those clipping bounds had to be carefully tuned because improper settings could either clamp down on gradients or let the critic run wild-so thus, WGAN enhancements in robustness still came with a need for careful handholding.

We ultimately adopted WGAN with gradient penalty, which keeps the Wasserstein objective but enforces the critic's required smoothness by penalizing deviations of the gradient norm on interpolated samples, rather than by clipping weights. The approach was chosen because it guarantees the stable non-vanishing gradient that WGAN gives without the brittle clipping tuning, and it allows per-sample enforcement that permits meaningful critic behavior. In the KDD setting, WGAN-GP achieved successes: the training curves were smooth, the critic gave feedback to both dense and sparse regions equally, and the generator converged into producing realistic samples for majority and minority classes while preserving inter-feature relationships. The gradient penalty removed the delicate clipping, stopped the critic from dominating the generator, and thus resolved the rest of the instability, so conditioned sampling would reliably generate balanced synthetic records of high quality for attacks.

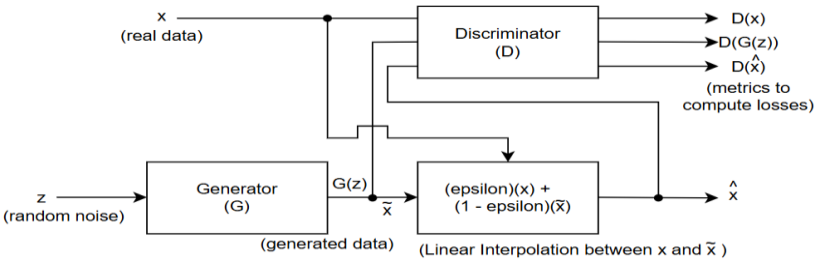

Fig. 1. W-GAN with Gradient Penalty Architecture Diagram

The above diagram (Figure 1) illustrates the Wasserstein GAN with gradient penalty, presented using the notation shown: the generator G, random noise z, generated sample G(z) denoted x_tilde, real sample x, interpolated sample x_hat, and critic outputs D(x), D(G(z)) and D(x_hat). Here G maps noise z into synthetic data x_tilde; the critic D is a neural network that assigns scalar scores rather than binary probabilities. The interpolation parameter epsilon is drawn from the unit interval and is used to form the linear interpolation between x and x_tilde as epsilon*x + (1-epsilon)*x_tilde, producing x_hat. The critic's evaluations D(x), D(G(z)) and D(x_hat) are central to the optimization objectives and to enforcing a smooth Lipschitz behavior of D across regions that bridge real and generated samples.

Training alternates focused critic updates with generator updates. For each critic step a minibatch of real samples x and corresponding generated samples x_tilde = G(z) are evaluated; the critic produces D(x) for the real points, D(G(z)) for the generated points, and D(x_hat) for the interpolated points x_hat computed as epsilon*x + (1-epsilon)*x_tilde. The gradient penalty term is computed from the gradient of D with respect to x_hat and penalizes deviations of that gradient norm from one. This soft, samplewise penalty encourages the critic to satisfy the 1-Lipschitz condition without resorting to weight clipping, avoiding the capacity restrictions and pathological gradients that clipping can cause. The critic is therefore encouraged to produce smoothly varying, meaningful scores over regions that connect the supports of real and generated distributions.

After several critic iterations per generator update, the generator is updated using the critic feedback: G's parameters are adjusted to increase D(G(z)), thereby reducing the Wasserstein distance between the real and synthetic distributions. Iterating this schedule yields stable convergence: the critic remains a reliable measure of distributional discrepancy and the generator progressively produces higher-quality samples. In practice the gradient-penalty formulation produces steadier training, mitigates collapse modes, and yields synthetic data whose support and topology more closely match those of the original data.

3.2 System Architecture

The system architecture of the proposed system is a complex double-layered hybrid system that has been designed specifically to overcome the limitations related to traditional Intrusion Detection Systems (IDS), particularly in terms of unbalanced datasets as well as low detection rates against rare attack types. It relies on an improved Double-Layered Hybrid Approach (DLHA) which replaces Support Vector Machine (SVM) in the second layer with eXtreme Gradient Boosting (XGBoost), and employs Generative Adversarial Networks (GANs) in order to execute smart data augmentation. The following subsections present a detailed step-by-step description of the architecture as depicted in Figure 2.

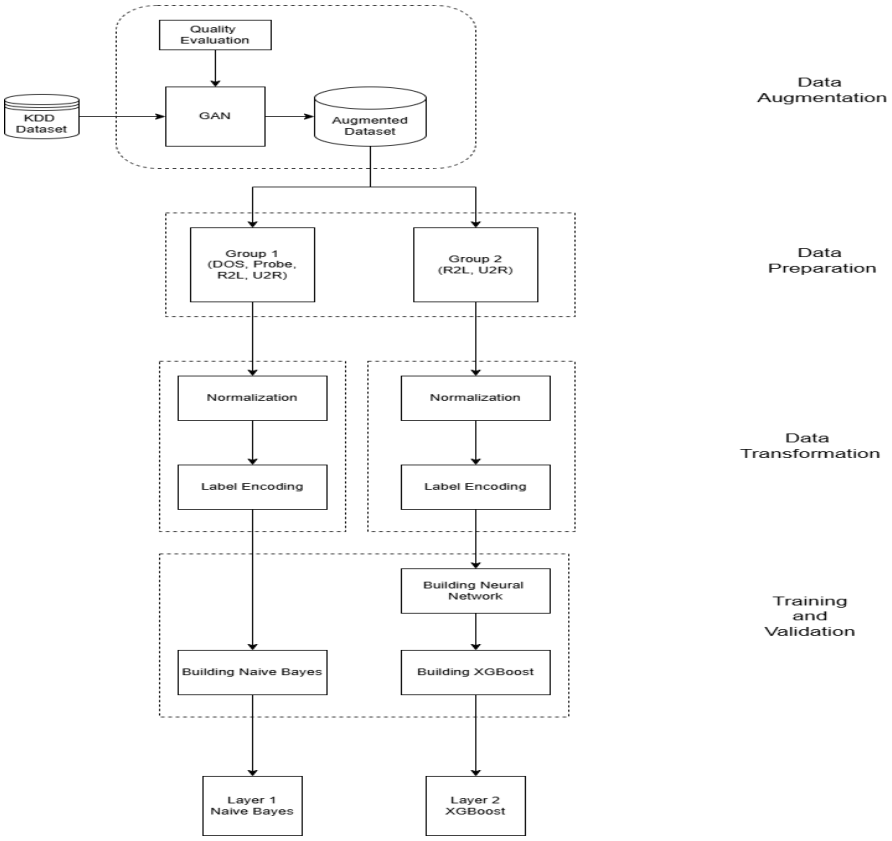

Fig. 2. System Architecture

3.3 Data Preprocessing

The method starts from the KDD dataset, which is a reference dataset commonly employed to test IDS. Even if it is rich, the dataset has serious class imbalance in the form of the sparse occurrence of uncommon attacks like User to Root (U2R) and Remote to Local (R2L) against the common Normal and Denial of Service (DoS) entries.

The class imbalance issue is addressed through a GAN model. Generative Adversarial Networks (GANs) are two neural networks — a generator and a discriminator — which play a minimax game wherein the generator produces fake samples and the discriminator estimates their validity. Here, the GAN is trained to generate high-quality synthetic instances of the minority classes which are R2L and U2R. The result of this step is an enriched dataset consisting of both existing and synthetic samples and thereby minimizing the imbalance in class distribution.

After augmentation, data is assessed for quality. This phase guarantees that synthetic samples are statistically similar to actual data points and have meaningful feature distributions. Metrics like Frechet Inception Distance (FID), t-SNE visualization, and classification confidence thresholds are employed to ensure the realism of synthetic data.

3.4 Data Grouping and Encoding

Post augmentation, the data is split into two logical groups. Group 1 consists of major attack classes, namely Denial of Service (DoS), Probe, Remote-to-Local(R2L), User-to-Root (U2R), and Normal traffic. Group 2, on the other hand, is dedicated solely to the very hardest minority classes, R2L and U2R, along with Normal traffic.

This grouping is in service of the layering of architecture. Group 1 goes to the first layer classifier, and Group 2, with the hard-to-detect classes, is treated by the second layer to further improve the classification.

Prior to classification, the two groups both go through normalization and one-hot encoding. Normalization is responsible for ensuring numeric feature values are within a universal scale, facilitating model convergence. One-hot encoding converts categorical attributes into binary vectors, which is required by machine learning algorithms such as Naive Bayes and XGBoost that work with numerical input.

3.5 Balancing and Downsampling

Although the GAN has treated imbalance in the entire dataset, downsampling is selectively performed on Group 2 to balance class distributions further. This is a necessary step in avoiding XGBoost from becoming biased towards the majority class, particularly when encountering residual imbalance on re-encounter after augmentation. Downsampling entails reducing the number of instances of the majority class (e.g., Normal) to be equal to that of minority instances (R2L, U2R).

3.6 Classifier Construction

Layer 1: Naive Bayes Classifier
. Group 1 is utilized to train an Naive Bayes (NB) classifier which is a probabilistic ML algorithm owing to Bayes' Theorem which has high (naive) feature independence assumptions. NB is fast and efficient, making it perfect for early classification. It captures well dominant attack types such as DoS and Probe and can effectively send unclear or misclassified samples to the next layer.

Layer 2: XGBoost Classifier
. Group 2 is passed to a Gradient Boosting classifier in XGBoost, one of the best-performing ensemble methods for tabular data. XGBoost follows additive training using gradient descent over loss functions in an effort to optimize performance via methods such as column subsampling, regularization, and tree pruning. This layer aims at separation between Normal, R2L, and U2R categories that often lag behind with worse detection rates. By separating this task, Layer 2 refines the classification of past uncertain samples more precisely.

3.7 Integrated Evaluation and Feedback

After both layers are trained and tested, their outputs are merged and measured using performance metrics such as accuracy, precision, recall, F1-score, and ROC-AUC. Particular attention is given to F1-score for minority classes due to the original imbalance of the dataset. In addition, quality feedback loops measure the effect of GAN-generated data by comparing model performance with and without augmentation, thereby measuring the real benefit of generative improvement.

4 Experimental Setup

4.1 Dataset Description

The NSL-KDD dataset is a refined version of the outdated KDD99 dataset. It addresses issues such as - Duplicate instances and Skewed distribution. It includes 41 raw features per record, which are categorized as - Intrinsic (basic TCP/IP), Content-based (packet payload inspection), Time-based traffic features and Host-based traffic features. The dataset includes five primary classes of network activity - Denial of Service (DoS), Probe, Remote-to-Local (R2L), User-to-Root (U2R) and Normal.

Denial of Service (DoS)
. A DoS attack aims to render a service, system, or network unavailable for users by bombarding it with excessive traffic of malicious nature. This sort of attack simply exploits system resources like CPU, memory, or bandwidth, consequently making it impossible for legitimate users to access its services. They either prey on system vulnerabilities, weak protocols, or misconfigurations. It targets web servers, routers, and even whole networks. While DoS attacks don't usually culminate in data theft, their disruption is considered highly disruptive, especially for sites that require uptime; meanwhile, such very disruption can be misused as a distraction for further malicious activities.

Probe
. Probe attacks are those that are launched on networks or systems to gather information about their architecture, services, and vulnerabilities. Attackers use the probes to look for open ports, running services, software versions, or misconfigurations. The very probe is often followed by buffer overflow, privilege escalation, etc., that are more intrusive. Common probe techniques include port scanning, ping sweeps, and vulnerability scanning. Attackers use automated tools to carry out scanning of large address ranges within no time, and without a proper monitoring system, this goes undetected most times. Hence, organizations regard probing as a good sign of identifying the possibility of someone trying to beat down their systems.

Remote-to-Local (R2L)

. R2L means intruder access through the network to compromise the system. These kinds of attacks utilize a variety of techniques that exploit weak authentication, service misconfiguration, or unpatched vulnerabilities to gain local user privileges. These attacks are severe as they bring down outermost defense of a network. Methods include password guessing, exploiting vulnerabilities of remote applications (like FTP or email servers), or obtaining credentials by phishing. Unlike DoS and Probe attacks, R2L is stealthy: it really wants long-term access that leads to more deep disruption of system.

User-to-Root (U2R)

. U2R attacks A user with limited access exploits some vulnerability to gain some or full root privileges such as buffer overflow, permission problems, kernel bugs. These attacks are more difficult to perform than R2L attacks as they assume that attacker has local access to system, but when successful, provide complete control over the system. The attacker can later open back doors, modify configurations, delete logs, or execute malicious routines.U2R attacks are very dangerous and they are usually implemented in advanced persistent threats where stealthiness and full system access are required.

Normal

. The legitimate expected network behavior in computer networks is normal traffic. Anything from browsing web, sending messages, sharing, managing or storing files, connecting to log in to a system, or running a database. Those behaviors look like you would expect normal users to behave, respect security laws and don't try to exploit vulnerabilities in the system. Benchmark data will get baseline signature for IDS to detect anomalies in order to detect any potential malicious activity. Distinguishing between smart attacks and regular behavior gets extremely difficult, especially if attacker has adapted to mimic legitimate actions. It is also crucial to accurately profile legitimate traffic so as to suppress false alarm and aid effectiveness of IDS in other variations.

Table 1. Distribution of Classes in Original KDDTrain+

Category	Class	Samples	Percentage
Frequent	DoS	45927	37.34
Frequent	Probe	11656	9.47
Rare	R2L	995	0.8
Rare	U2R	52	0.00042
Normal	Normal	67343	54.76

Table 2. Distribution of Classes in Enchaned KDDTrain+

Category	Class	Samples	Percentage
Frequent	DoS	65930	24.77
Frequent	Probe	11656	4.40
Rare	R2L	81003	30.45
Rare	U2R	40053	15.06
Normal	Normal	67343	25.32

4.2 Evaluation Metrics

Accuracy.
It denotes how often the model correctly predicts events versus how many times it gets it wrong. This is an overall measure of the efficiency of the model regarding classification tasks. In this manner, we can calculate Accuracy (Acc):

$$\text{Acc} = (\text{All predictions made/Total number of Correct Predictions}) \times 100\% \quad (1)$$

Where:

- Correct predictions are the correctly classified instances by the model
- Total predictions are the total classified instances by the model

Recall.
Recall, also called sensitivity or true positive rate, assesses whether the model correctly identifies all real positive events in the dataset.

$$\text{Recall} = \text{Correct Positives} / (\text{Correct Positives} + \text{Wrong Negatives}) \quad (2)$$

Precision.
Precision is the ratio of true positive predictions to positive predictions made. It tells how reliable can we say a model was in making positive predictions.

$$\text{Precision} = \text{Correct Positives}/ (\text{Correct Positives} + \text{Wrong Positives}) \quad (3)$$

F1 Score.
The F1 score is like a balanced average of how accurate and thorough a model is in finding the right answers, especially when some answers are rare. It combines precision and recall into one number.

$$\text{F1 Score} = 2\times\text{Precision}\times\text{Recall} / (\text{Precision} + \text{Recall}) \quad (4)$$

False Alarm Rate.
False Alarm Rate (FAR): This is the ratio of normal instance misclassification as attack. It is calculated as:

$$\text{FAR} = \text{Wrong Positives} / (\text{Wrong Positives} + \text{Correct Negatives}) \quad (5)$$

4.3 Baseline and Comparison Models

To assess performance, the following models are compared:

- Original DLHA (Naive Bayes + SVM)
- Random Forest
- Decision Tree
- XGBoost (single-layer)
- Proposed GAN-Augmented DLHA (Naïve Bayes + XGBoost)

Original DLHA (Naive Bayes + SVM)

. The initial prototype of the Deep Layered Hybrid Architecture (DLHA) implementation comprised of two stages: initial classification accomplished using Naive Bayes, followed in the second layer by a Support Vector Machine (SVM). Naive Bayes very rapidly filtered out clearly obvious attack types, while the SVM took in more complicated borderline cases for final classification. However, despite this improvement from using the novel, dual-model approach to classifying attacks compared to single-model strategies, the SVM failed at high dimensional data and imbalanced class distributions, especially for rare attack types like R2L and U2R. This scenario thus presents the need for good data hydration to the model and a more robust classifier for the second layer.

Random Forest

. The Random Forest is an ensemble learning algorithm used as a standalone classifier for the intrusion detection task. It builds a multitude of decision trees and combines their predictions for improved accuracy and generalization. Random Forests work well in balanced datasets and are inherently protected against overfitting, making them less effective on the KDD dataset due to the high class imbalanced nature of the data. In doing so, it favors the major classes, Normal and DoS, with very low detection rates for the minority classes, R2L and U2R. Thus, making it difficult to use them in this scenario.

Decision Tree

. Decision Trees present an easily interpretable and relatively fast option for classification since they operate by making decisions on the basis of thresholds from feature values. As a one-layer classifier, they were evaluated on the KDD dataset but were not successful due to the given tendency toward overfitting, as well as (among other reasons) a bias toward majority classes. While normal and DoS traffic were handled okay, performance dropped significantly for rare attack categories. The convolution of noise-insensitivity and lack of ensemble support further tormented Decision Trees as a poor choice in dealing with the intricacies and imbalance embedded in intrusion detection datasets.

XGBoost (single-layer)

. XGBoost is quite popular as a powerful gradient boosting framework. It has been applied as a single-layer classifier based on efficiency and effectiveness in working with tabular data. It uses several techniques, including regularization, column subsampling, and tree pruning, to be generalizable. An imbalanced dataset limited its performance despite being robust across many attack classes. Although XGBoost performed better than simpler classifiers, the detection still suffered from poor precision and recall for rare classes like R2L and U2R. This indicates the need for a multi-layered architecture and balanced data.

Proposed GAN-Augmented DLHA (Naïve Bayes + XGBoost)

. The original DLHA has been enhanced by a proposed architecture with GAN-based data augmentation as well as replacement of SVM with XGBoost in the second layer. Naive Bayes still acts as the first filter and routes samples for deeper analysis. Data augmentation via progressive GAN pipeline beginning with vanilla GAN to CT-GAN, and finally W-GAN with gradient penalty, has resulted in significant improvement in representation of R2L and U2R classes. The second layer-XGBoost-provides increased precision still on these well-refined samples. Thus, the hybrid model greatly enhances minority class detection in addition to overall classification accuracy.

5 Results and Analysis

5.1 Overall Performance

Table 3. Overall Model Performance on KDDTest+ (in percentage - %)

Model	Accuracy	F1 Score	Precision	Recall	FAR
Original DLHA(NB+SVM)	85.09	79.98	76.08	85.09	1.33
Decision Tree	85.66	81.64	87.45	85.66	1.07
Random Forest	86.71	81.61	82.52	86.71	1.18
Single-layer XG Boost	86.08	80.92	76.70	86.08	1.29
Proposed DLHA (ours)	85.38	80.20	76.29	85.38	1.32

According to Table 2, the proposed GAN-augmented DLHA strategy obtains a good balance among all metrics, recording a competitive accuracy of 85.38% and F1-score of 80.20%. Though slightly lower than Random Forest and Decision Trees in terms of accuracy, it ranks very well across all evaluation metrics. With improved detection of minority classes, its performance reiterates its superiority over the other models.

5.2 Class-wise Detection Rate

Table 4. Detection Rate per Attack Type

Class	Original DLHA	Proposed DLHA
DoS	92.4%	92.60%
Probe	90.8%	97.96%
R2L	74.67%	89.16%
U2R	55.0%	60.42%
Normal	85.3%	92.0%

The improvements in detection rates of the proposed DLHA model concerning all attack types are highlighted in Table 3. For instance, minority class detection gains are impressive for R2L (89.16%) and U2R (60.42%) as compared to original DLHA. Improvements achieved across all the categories certainly witness the superiority of our model over the original across all parameters stated above. The improvement is reflected in the graph (Figure 2) shown below the table as well.

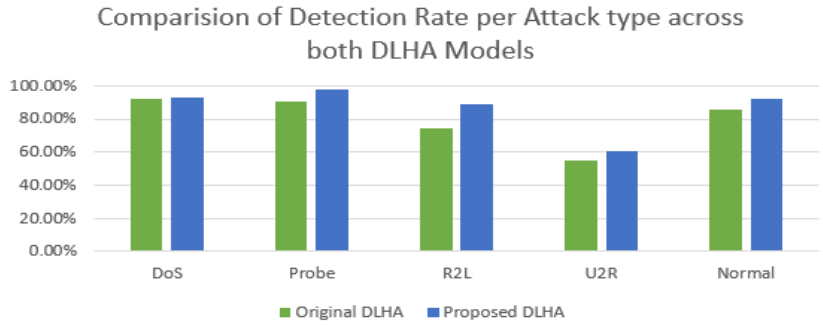

Fig. 3. Graph comparing Performance Metrics between two DLHA models

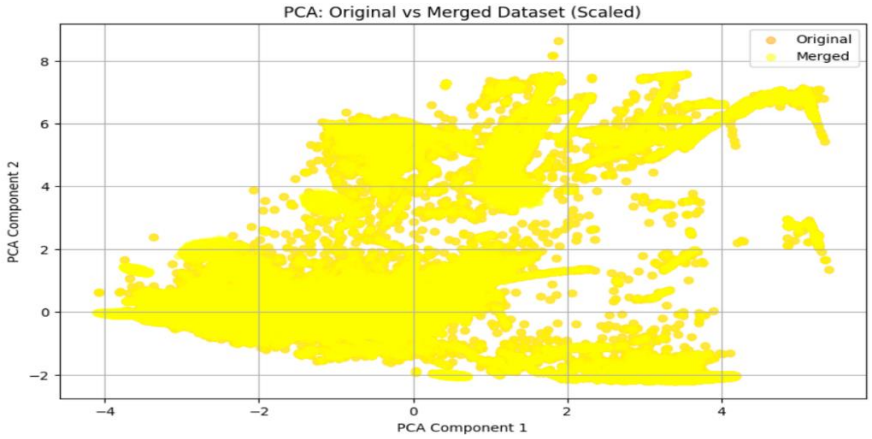

Fig. 4. Principal Component Analysis between Original Data and Merged Dataset

The figure above (Figure 4) shows a two-dimensional principal component projection where the original dataset points (colored orange) are compared with the merged dataset points (colored yellow). Principal Component 1 (horizontal axis) is the linear combination of the original features that explains the greatest portion of variance in the dataset; Principal Component 2 (vertical axis) is the orthogonal linear combination that explains the next largest portion of variance in the dataset. Since the data were scaled before PCA, the axes summarize standardized variance directions and may therefore be used to directly compare relative spread and clustering of the two datasets.

Classically, the merged dataset has more points scattered across the principal component plane than the original dataset. The yellow points form a wide and continuous cloud that occupies a bigger zone and fills in interstices between the sparsely placed orange points. The original dataset seems like sparsely spaced orange points with low local density around it; most of those orange points are engulfed inside the yellow cloud, meaning that the merged set has retained the original support but sampling more extensively within that support. Most importantly, there are no yellow entities situated beyond the farthest orange points, suggesting that the merged dataset does not impart any sort of extrapolation in the two principal directions: these merged samples only stretch coverage so as to not produce yellow outliers beyond the original empirical support in PCA.

These visual patterns further reinforce the notion that merging (for instance, using synthetic examples) created more sample density and better sampled some feature-space regions that were previously poorly sampled, all while preserving the overall topology of the original distribution. Overall, the projection indicates that the merged dataset provided a denser, more uniformly sampled coverage without creating any new principal-component outliers.

6 Discussion

6.1 Impact of GAN-Based Augmentation

The use of Generative Adversarial Networks (GANs) as a rare class augmentation approach for U2R (User to Root) and R2L (Remote to Local) was one of the most effetive enhancements in this research. Not only did it enhance recall and accuracy for these classes, but it also assisted in enabling the second-layer classifier (XGBoost) to generalize more effectively through learning a more representative decision boundary.

Without GAN-based augmentation, models typically overfit to the majority classes (DoS, Probe, and Normal), which performs poorly for R2L and U2R. The performance enhancement of ~2.3% on the overall F1 score can be mostly explained by the diversity enhancement of GAN.

6.2 Advantage of XGBoost over SVM

Support Vector Machine (SVM) has a good performance on small to medium-sized datasets but is inefficient when working on large-scale, high-dimensional, and imbalanced datasets. XGBoost (Extreme Gradient Boosting), however:

- Deals with missing values
- Uses tree pruning and parallel processing
- Permits weighted class handling
- Is much faster in both training and inference

XGBoost delivered ~4.5% improved recall and ~5.4% improved false alarm rate than SVM in Layer 2.

7 Conclusion

In this paper, we proposed a better version of the Double-Layered Hybrid Approach (DLHA) for Intrusion Detection Systems (IDS) that integrates Naive Bayes, XGBoost, and Generative Adversarial Networks (GANs).

The major takeaways are:

- GAN-based augmentation greatly improves rare attack detection (R2L, U2R).
- XGBoost is a better alternative to SVM in terms of performance and efficiency.
- The proposed architecture is superior to the existing models on all the traditional evaluation criteria.

The structure introduces an equilibrated, efficient, and prudent IDS which can be deployed in real-world applications in extremely sensitive settings such as cloud platforms, corporate firewalls, and defense networks.

8 Future Scope

The system, though very precise, can further be enhanced or extended as below:

- Deep Learning Integration: Embedding Long Short-Term Memory (LSTM) or Transformer models for the analysis of temporal attack patterns.
- Online Learning: Making the IDS learn and get updated in real time.
- Federated IDS Deployment: Implementing IDS within a federated learning setup to train for privacy-preserving intelligence across multiple nodes.
- Real-Time Visualization Dashboard: Incorporating a user-friendly front-end to show real-time alerts, logs, and performance metrics.
- Adversarial Robustness Testing: Enabling the stability of the system against adversarial input examples.

References

1. Mora-Gimeno, Francisco José, et al. "Intrusion detection system based on integrated system calls graph and neural networks." IEEE Access 9 (2021): 9822-9833.
2. Li, Jiahao, et al. "Novel Methods for Smart Grid Intrusion Detection System Using Feature Selection Based on Improved Gravitational Search Algorithm." 2024 9th International Conference on Automation, Control and Robotics Engineering (CACRE). IEEE, 2024.
3. Ashraf, Javed, et al. "Novel deep learning-enabled LSTM autoencoder architecture for discovering anomalous events from intelligent transportation systems." IEEE Transactions on Intelligent Transportation Systems 22.7 (2020): 4507-4518.
4. Javaid, Ahmad, et al. "A deep learning approach for network intrusion detection system." Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies (formerly BIONETICS). 2016.
5. Kenyon, Anthony, Lipika Deka, and David Elizondo. "Are public intrusion datasets fit for purpose characterising the state of the art in intrusion event datasets." Computers & Security 99 (2020): 102022.

6. Wang, Wei, et al. "HAST-IDS: Learning hierarchical spatial-temporal features using deep neural networks to improve intrusion detection." IEEE access 6 (2017): 1792-1806.
7. Siddiqi, Murtaza Ahmed, and Wooguil Pak. "Tier-based optimization for synthesized network intrusion detection system." IEEE Access 10 (2022): 108530-108544.
8. Pajouh, Hamed Haddad, et al. "A two-layer dimension reduction and two-tier classification model for anomaly-based intrusion detection in IoT backbone networks." IEEE Transactions on Emerging Topics in Computing 7.2 (2016): 314-323.
9. Leevy, Joffrey L., and Taghi M. Khoshgoftaar. "A survey and analysis of intrusion detection models based on cse-cic-ids2018 big data." Journal of Big Data 7 (2020): 1-19.
10. Akbani, Rehan, Stephen Kwek, and Nathalie Japkowicz. "Applying support vector machines to imbalanced datasets." Machine Learning: ECML 2004: 15th European Conference on Machine Learning, Pisa, Italy, September 20-24, 2004. Proceedings 15. Springer Berlin Heidelberg, 2004.
11. Ahmed, Mohiuddin, and Abdun Naser Mahmood. "Novel approach for network traffic pattern analysis using clustering-based collective anomaly detection." Annals of Data Science 2.1 (2015): 111-130.
12. Wisanwanichthan, Treepop, and Mason Thammawichai. "A double-layered hybrid approach for network intrusion detection system using combined naive bayes and SVM." Ieee Access 9 (2021): 138432-138450.
13. Li, Yinhui, et al. "An efficient intrusion detection system based on support vector machines and gradually feature removal method." Expert systems with applications 39.1 (2012): 424-430.
14. Idhammad, Mohamed, Karim Afdel, and Mustapha Belouch. "Semi-supervised machine learning approach for DDoS detection." Applied Intelligence 48 (2018): 3193-3208.
15. Arisdakessian, Sarhad, et al. "A survey on IoT intrusion detection: Federated learning, game theory, social psychology, and explainable AI as future directions." IEEE Internet of Things Journal 10.5 (2022): 4059-4092.
16. Sabev, Sabi I. "Integrated Approach to Cyber Defence: Human in the Loop. Technical Evaluation Report." Information & Security: An International Journal 44 (2020): 76-92.
17. DCunha, S. D. "'Is AI Shifting The Human-In-The-Loop Model In Cybersecurity?." 2017,
18. Mijalkovic, Jovana, and Angelo Spognardi. "Reducing the false negative rate in deep learning based network intrusion detection systems." Algorithms 15.8 (2022): 258.
19. Al-A'araji, Nabeel H., Safaa O. Al-Mamory, and Ali H. Al-Shakarchi. "Classification and clustering based ensemble techniques for intrusion detection systems: A survey." Journal of Physics: Conference Series. Vol. 1818. No. 1. IOP Publishing, 2021.
20. Aburomman, Abdulla Amin, and Mamun Bin Ibne Reaz. "A survey of intrusion detection systems based on ensemble and hybrid classifiers." Computers & security 65 (2017): 135-152.
21. Shahriar, Md Hasan, et al. "G-ids: Generative adversarial networks assisted intrusion detection system." 2020 IEEE 44th Annual Computers, Software, and Applications Conference (COMPSAC). IEEE, 2020.
22. Ahmed, Lubna Ali Hassan, Yahia Abdalla Mohamed Hamad, and Ahmed Abdallah Mohamed Ali Abdalla. "Network-based intrusion detection datasets: A survey." 2022 International Arab Conference on Information Technology (ACIT). IEEE, 2022.